REMARKS

This submission accompanies a Request for Continued Examination and is in response to the Advisory Action dated August 15, 2006.

Claims 1-9 and 25-33 were pending as of the date of the Advisory Action. New claims 34-39 have been added. Claims 1, 5, 6, 25, 29 and 30 are amended herein.

The Advisory Action maintains the rejection of claims 1, 2, 9, 25, 26 and 33 under 35 U.S.C. § 103(a) as being unpatentable over the article by Bhattacharya et al. ("Bhattacharya") in view of U.S. Patent No. 6,785,690 ("Davidson"), and it maintains the rejection of claims 3-8 and 27-32 under 35 U.S.C. § 103(a) as being unpatentable over Bhattacharya and Davidson and further in view of U.S. Patent 6,070,174 ("Starek"). While Applicants believe that the claims as presented in response to the previous Office Action are patentable over the cited references, in the interests of moving prosecution forward, Applicants have amended the claims herein to further emphasize the differences between the claimed invention and the teachings of the cited references. Applicants have addressed these important differences below.

I. The Claimed Invention *Designates* A Field Of User Defined Type As Containing Data To Be Stored Outside Of The Database Store *Without Altering The Underlying Data Type Of The Field*, As Recited In Claims 1 and 25

Claims 1 and 25 have been amended to recite more clearly that each of the plurality of fields of the user-defined type is "assigned any one of a plurality of data types supported by the database store" and that "at least one of said plurality of fields of the definition [is] *designated* as containing data that is to be stored as a file outside of the database store separately from the data of the other of said plurality of fields ... *without altering the assigned data type of said at least one designated field.*" *See, e.g.*, claim 1 (emphasis added). Additionally, both claims now recite the step of "determining from the *designation* of said at least one field that the data of that field is to be stored as a file outside of the database store *irrespective of the assigned data type of that field.*" This feature is illustrated by way of example in the specification.

In the example of Figure 6, a "PhotoFS" field of a user-defined type called "Person" is assigned the data type "SqlBytes," which is a native data type supported by the database store. According to the invention, the "PhotoFS" field is "designated" as a field whose data

is to be stored outside of the database by use of a separate attribute added to the field definition. Specifically,

> this is accomplished by annotating the field of the CLR class definition of the UDT with a custom attribute that provides the designation. In particular, a new property of the SqlUdtField() custom attribute described above in the Background of the invention has been created. The new property is called "IsFilestream." A value of "true" for that property (*e.g.*, IsFilestream = true) indicates that this field of the Person type shall be stored as a file outside of the database store in accordance with the present invention. . . . The annotations to the fields of a user defined type, including the IsFilestream property of the SqlUDTField() custom attribute, define a set of metadata associated with the defined type.

Specification, ¶ 0057. While in the example shown, the "IsFilestream" property is used to designate a field (*i.e.*, the "PhotoFS" field) having the "SqlBytes" data type as containing data to be stored outside of the database store, "the property may be applied to fields of other data types, as desired." Spec, ¶ 0058. Because the underlying data type of the field is not altered, applications are able to work against the data of the designated field in the same way as they would access the data of that data type if stored inside the database. For example, a field having the character string or binary string data type can be designated to be stored as a file outside of the database store, but database functions that operate on those data types, such as the *substring*() function, can still be applied to the data even though it's stored as a file outside the database. None of the cited references teach or suggest this feature of the invention.

In particular, the "Datalink" feature described in the Battacharya article is a new data type in itself. "DataLinks functionality is supported by introducing DATALINK as an SQL data type in the DBMS ...." Battacharya, p. 501, § 2. It defines its own access application programming interfaces (APIs) and programming model, but contrary to the claimed invention, it does not allow fields of any other existing data types to be annotated to designate them as containing data to be stored outside the database while allowing applications to continue to work against the data of those designated fields in the same way as they would if those data types were stored inside the database. This represents a significant and fundamental difference between the claimed invention and the Datalinks feature. Applicants claimed invention is much more than simply providing a link from a column entry of a database to a file outside the database. Because of this fundamental difference between the

claimed invention and the Datalinks feature of Battacharya, Applicants respectfully submit that independent claims 1 and 25 patentably define over Battacharya, alone or in combination with any of the other cited references. None of the other cited references cures the deficiencies of Battacharya. Reconsideration of the rejection of independent claims 1 and 25 is therefore respectfully requested.

II. Other Dependent Claims Also Recite Features That Distinguish Over The Cited Art Of Record

Inasmuch as the remaining claims all depend either directly or indirectly from independent claim 1 or 25, Applicants submit that they too patentably define over the cited art for the same reasons. Nevertheless, Applicants submit that the features recited in various dependent claims further lend patentability to those claims for the reasons presented below.

A. New Claims 35, 36, 38 and 39

New claims 35 and 38 recite the additional step of:

> performing a database operation on the data of said at least one designated field of the instance of the user-defined type, wherein the database operation is performed on the data of said at least one designated field as if it were stored within the database store.

See, e.g., claim 35. Claims 36 and 39 define the "database operation" as one of an "INSERT, UPDATE or DELETE" operation. As explained in the specification,

> It is important to note that even though the data for the corresponding Filestream fields [*i.e.*, the designated fields] of the instance of the UDT are stored as files outside of the database store, they are subject to the operations of the database engine as if stored within the database table. For example, as embodied in the SQL SERVER database engine, the T-SQL commands INSERT and UPDATE can be used to insert new data or update existing data into a file that stores the data of a Filestream field of an instance of a UDT, just as if the data field were stored within the database table. Similarly, the T-SQL DELETE command can be used to delete a row containing a UDT that has one or more Filestream fields stored in separate files; deleting the row deletes the referenced files as well. Filestream fields within a UDT that are stored in separate files can also be queried like any other column.

Spec., ¶ 0065. On the contrary, the Datalink feature of Battacharya does not appear to allow conventional database operations to be performed against data that is stored in a file outside of the database. Datalink simply provide a mechanism to link a column entry of a database

table to an external file. Nor do any of the other cited references teach or suggest this claimed feature. For this additional reason, Applicants respectfully submit that new claims 35, 36, 38 and 39 patentably define over the cited references.

B. Claims 3 and 27

Claims 3 and 27 recite that the data of the fields of the object that are stored ***within*** the database store are stored as fragments within a column of a database table that has been designated as having the user defined type. The last Office Action asserts that Starek teaches this feature at column 10, lines 29-51. Applicants do not understand how. That portion of Starek describes a "master file table" of an operating system that stores "file records" for the files stored in the operating system's file system. Claims 3 and 27 have nothing to do with files or file systems. Consequently, Applicants do not understand how combining this aspect of Starek with Davidson and Bhattacharya would produce the invention recited in claims 3 and 27. Moreover, because claims 3 and 27 have nothing to do with files or file systems, the discussion in the last Office Action of a "motivation" for combining Starek with Davidson and Bhattacharya, which focuses on file system features, does not make sense. Reconsideration of the rejection of claims 3 and 27 is respectfully requested for these additional reasons.

C. Claims 4 and 28

Claims 4 and 28 recite that "a unique identifier associated with the object" is stored "in another column of the table in a same row as the data of the fields of the object." These claims are directed to the "row_guid" identifier described in the specification at paragraphs 0060 and 0064. As described, the "row_guid" identifier is a *logical* identifier for a file that contains data of a designated field of an instance of a user-defined type, as opposed to an actual physical file name. When an application performs file system access to a file that contains the data of a field of an instance of a user-defined type, the logical row_guid identifier is converted to a physical path before accessing the file. This enables physical data re-organization (such as renaming the file to a different location) to be performed without interfering with the validity of the logical path.

The last Office Action asserts that this feature is taught by Bhattacharya's description of a "*RecoveryID_at_link*" identifier. Applicants respectfully disagree. Battacharya's "*RecoveryID_at_link*" field serves a different purpose; it is used mostly for recovery consistency. *See* Battacharya, at p. 503. Applicants' claimed identifier is not used for recovery consistency. Moreover, as the Battacharya article explains, the reference to the file is physical – as opposed to the logical reference provided by the claimed "row_guid" identifier. Specifically, in the example given in the paper, the "*RecoveryID_at_link*" field composes http as URL scheme, server name, and *actual* filename. Furthermore, there is no indication in Bhattacharya that the "*RecoveryID_at_link*" identifier is associated with an object, *i.e.*, an instance of a user defined type, that has been stored in a column of a database table, as claimed by Applicants. Applicants respectfully submit, therefore, that Bhattacharya does not teach the claimed identifier. For that reason alone, the rejection of these claims should be withdrawn.

Moreover, to the extent that this rejection again requires the combination of multiple references, the Examiner is required to state a motivation for combining the *RecoveryID_at_link* feature of Bhattacharya with the other references. In this case, the Examiner has provided **no evidence of any motivation whatsoever**. Consequently, the rejection is improper and must be withdrawn.

D. Claims 5 and 29 and New Claims 34 and 37

Claims 5 and 29 have been amended to recite the additional steps of:

> creating a unique directory within a file system of the computer for storing files containing the data of said at least one designated field of every instance of the user defined type; and
>
> storing the data of said at least one designated field of every instance of the user defined type as a respective file within the created directory

See, e.g., claim 5. That is, for each instance of the user defined type that is stored, the data in the designated field of that instance is stored as a file in a file system directory that is uniquely created for the user defined type. Claims 34 and 37 recite the additional step of creating such unique directories for each of multiple, different user-defined types. That is, as illustrated in Figure 7 and described in the specification, "each UDT is assigned a different column-level directory within the computer file system." Spec., ¶ 0064. None of the cited

references teach or suggest these features, and therefore, Applicants submit that these claims recite patentable subject matter for this additional reason.

E. Claims 7 and 31

Claims 7 and 31, which depend from claims 6 and 30, respectively, recite the steps that are performed when an application requests access, via the file system of a computer, to the file in which the data of a designated field of a user-defined type is stored outside the database store. In particular, while access is made via the file system, these claims recite that the request for access to the file identifies the file *not* by its identity within the file system, but by identifying "the field of the object [for which the file holds data] ... within the *database store*," and then "determining from the identity of the field of the object within the database store a path within the file system of the computer to the file containing the data of that field of the object." *See, e.g.,* claim 7. As explained in the specification,

> applications are also able to access such files directly via the file system of the computer. Specifically, an application can generate a call, via the application programming interface to the file system, to open such a file directly. The call will identify the corresponding Filestream field of the object by its identity within the database store. Based on the identity of the field of the object within the database store, a file system path to the file containing the data of that field is determined. The call to open the file is then executed using the determined path.

Spec., ¶ 0068.

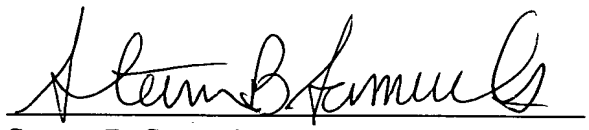
The last Office Action asserts that Starek teaches these claimed steps at column 7, lines 9-20 and column 4, lines 23-32. But none of these portions of Starek describe accessing a file via a file system, where the file is identified *not* by its identity in the file system, but rather by identifying the field of an object within a database store for which the file holds data. Because Starek does not teach the claimed feature, the rejection should be withdrawn.

Moreover, to the extent that this rejection again requires the combination of multiple references, the Examiner has again failed to provide ***any*** evidence of a motivation to make the asserted combination. For this additional reason, the rejection is improper and must be withdrawn.

CONCLUSION

For the foregoing reasons, Applicants respectfully submit that all of the claims patentably define over the cited art of record, and therefore, that the instant application is in condition for allowance. Reconsideration of the Advisory Action of August 15, 2006 and the prior May 30, 2006 Office Action is respectfully requested.

Date: October 30, 2006



Steven B. Samuels
Registration No. 37,711

Woodcock Washburn LLP
One Liberty Place - 46th Floor
Philadelphia PA 19103
Telephone: (215) 568-3100
Facsimile: (215) 568-3439